Domain-Adversarial Training of Neural Networks

Yaroslav Ganin ganin@skoltech.ru
Evgeniya Ustinova evgeniya.ustinova@skoltech.ru
Skolkovo Institute of Science and Technology (Skoltech)
Skolkovo, Moscow Region, Russia

Hana Ajakan hana.ajakan.1@ulaval.ca
Pascal Germain Pascal.Germain@ift.ulaval.ca
Département d'informatique et de génie logiciel, Université Laval
Québec, Canada, G1V 0A6

Hugo Larochelle hugo.larochelle@usherbrooke.ca
Département d'informatique, Université de Sherbrooke
Québec, Canada, J1K 2R1

François Laviolette Francois.Laviolette@ift.ulaval.ca
Mario Marchand Mario.Marchand@ift.ulaval.ca
Département d'informatique et de génie logiciel, Université Laval
Québec, Canada, G1V 0A6

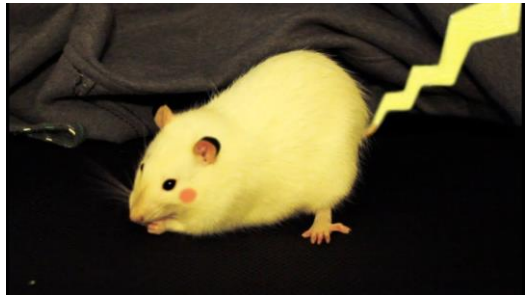
Victor Lempitsky lempitsky@skoltech.ru
Skolkovo Institute of Science and Technology (Skoltech)
Skolkovo, Moscow Region, Russia

Journal of Machine Learning Research 17 (2016) 1-35 Submitted 5/15; Published 4/16

Domain Adaptation

<Daytime, Source>

<Night, Target>



<Rat, Source>

<Pikachu, Target>

Domain Adaptation

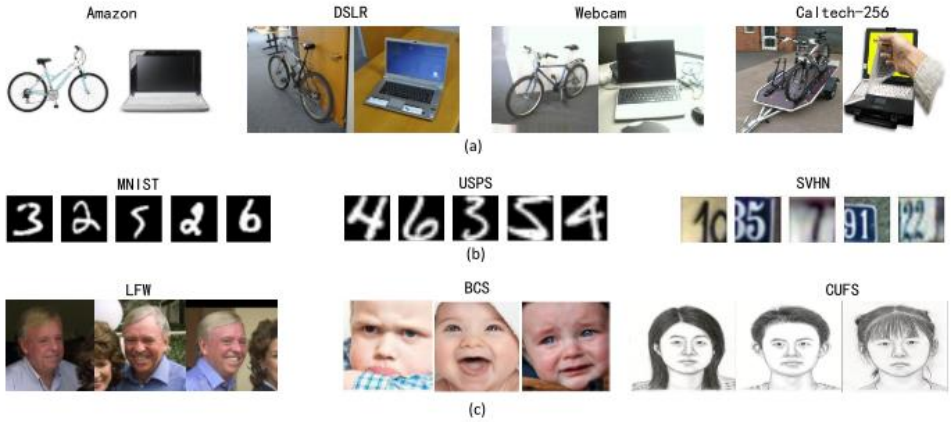
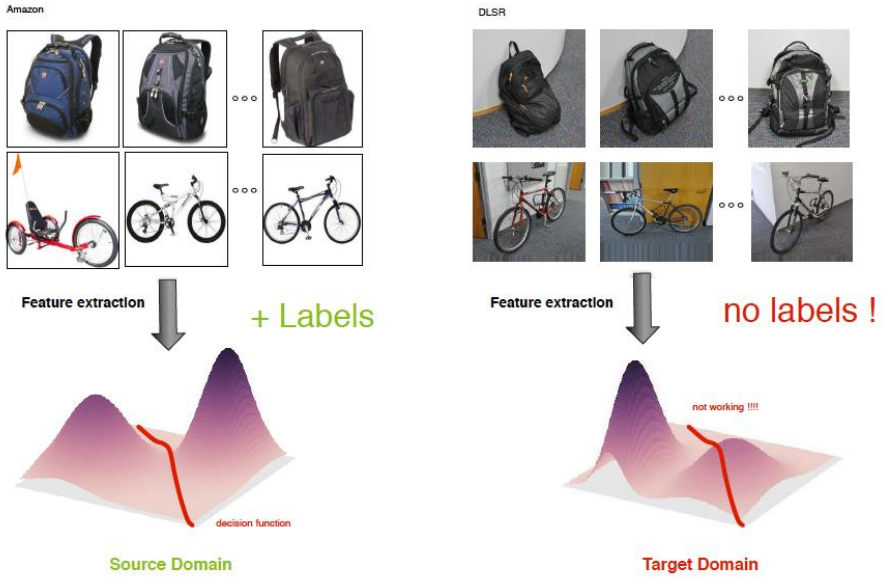
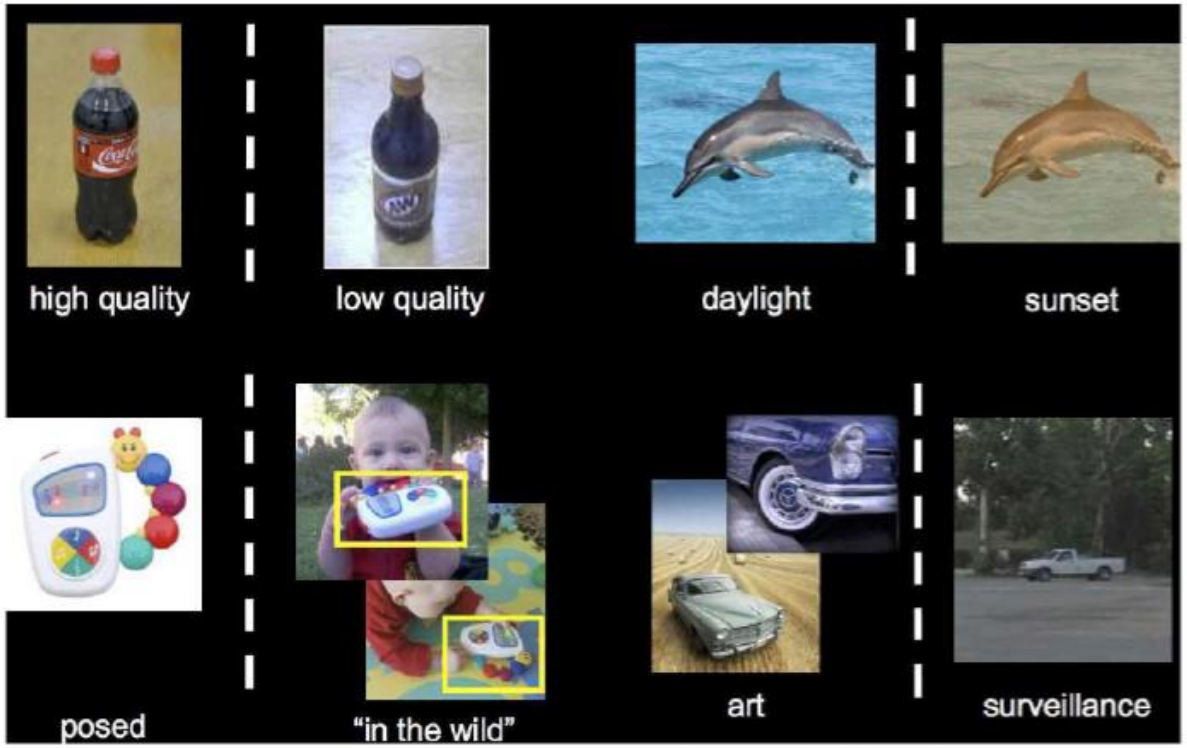
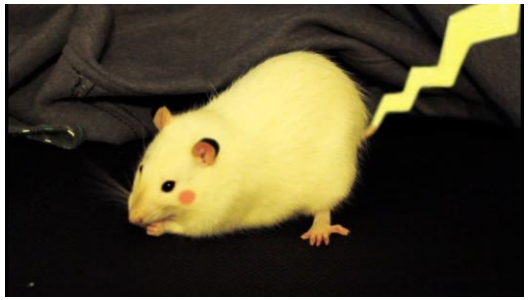


Fig. 1. (a) Some object images from the "Bike" and "Laptop" categories in Amazon, DSLR, Webcam, and Caltech-256 databases. (b) Some digit images from MNIST, USPS, and SVHN databases. (c) Some face images from LFW, BCS and CUFS databases. Realworld computer vision applications, such as face recognition, must learn to adapt to distributions specific to each domain.

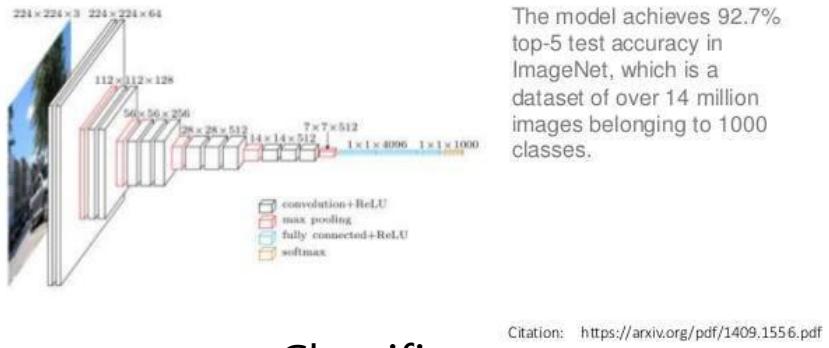




Domain Adaptation

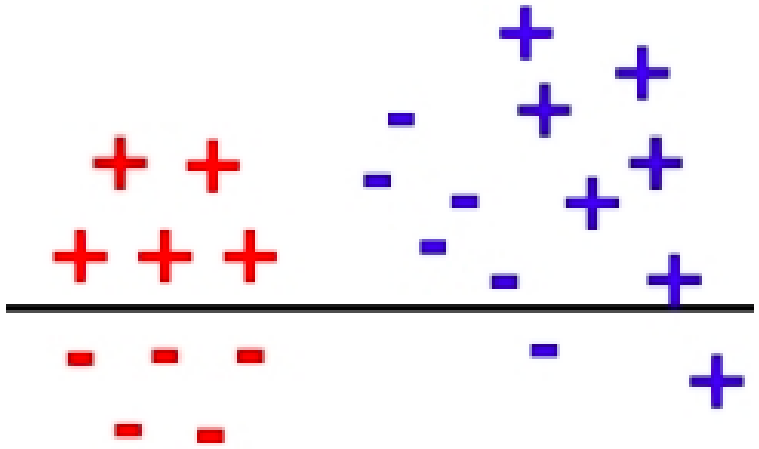


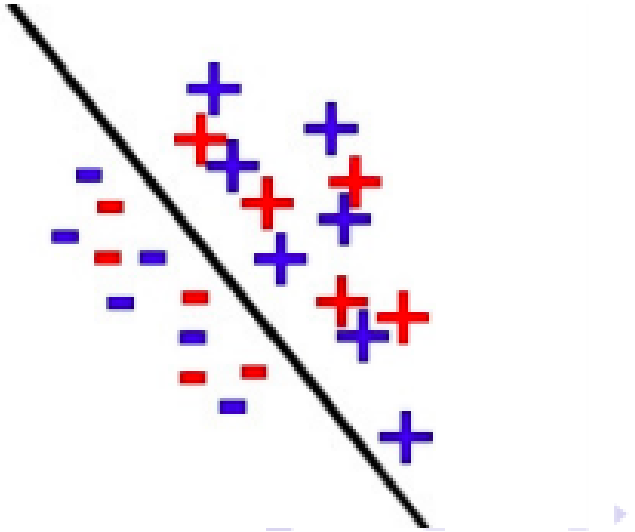
<Rat, Source>



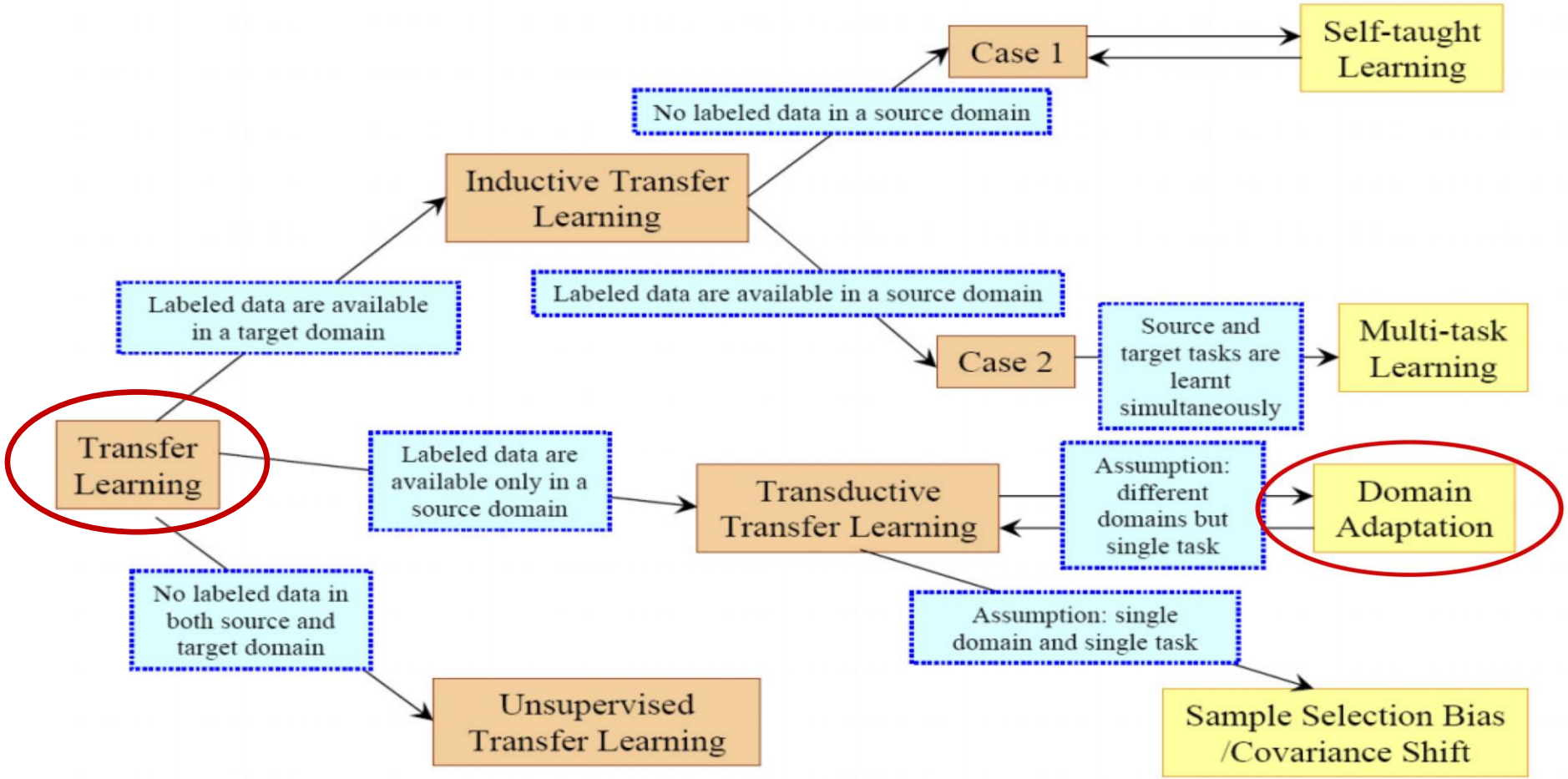
<Classifier, η>

<Pikachu, Target>





Domain Adaptation



Domain Adaptation

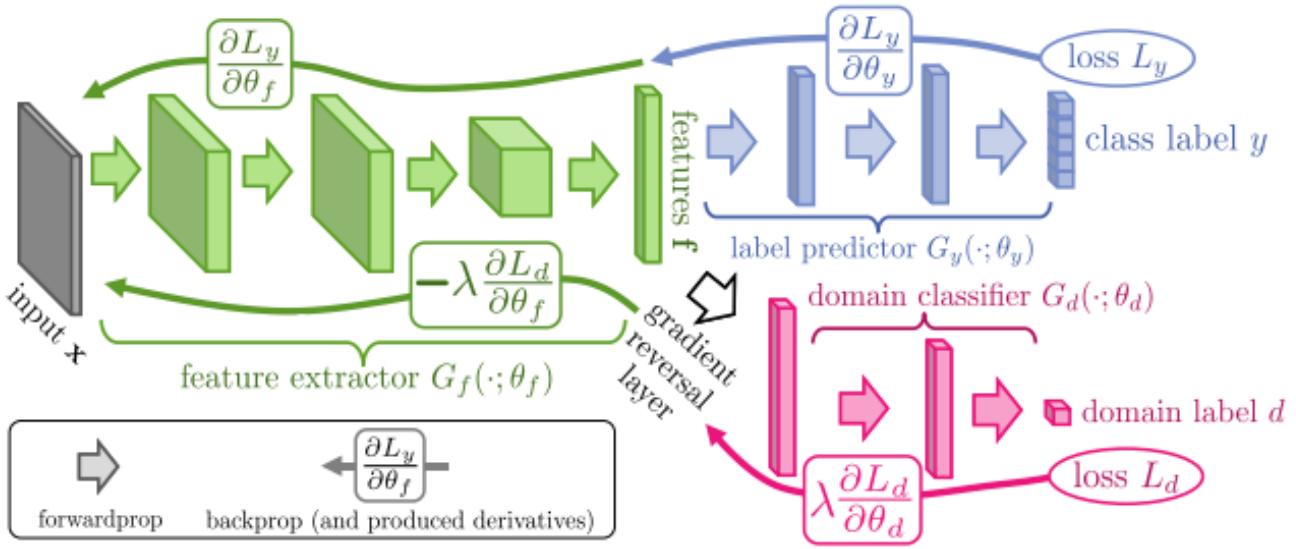
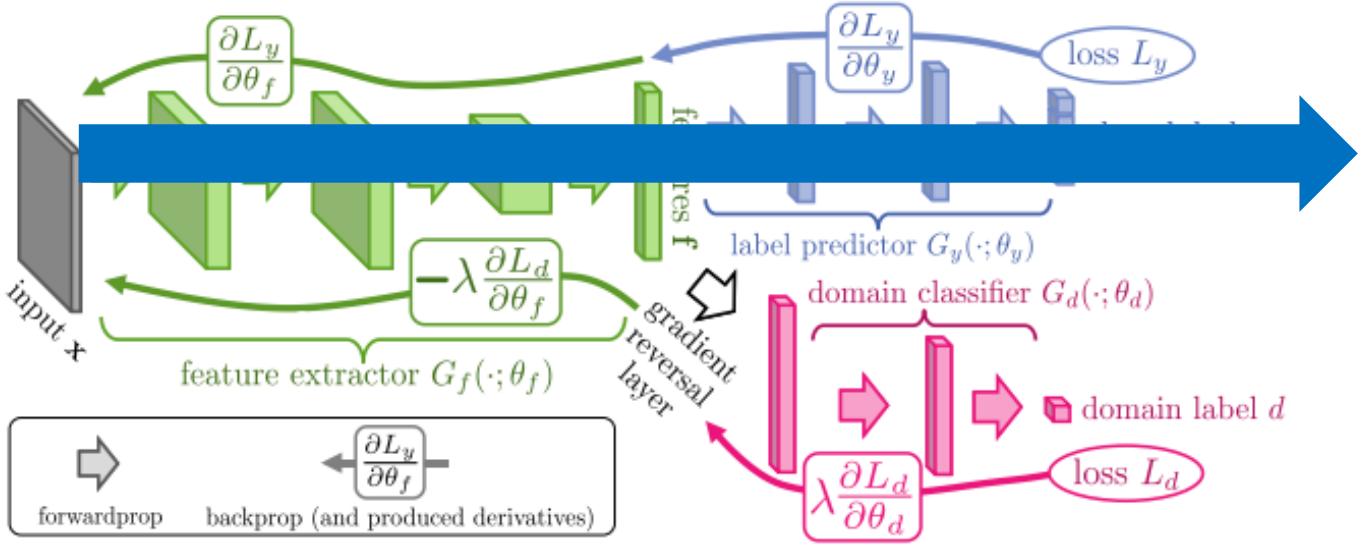
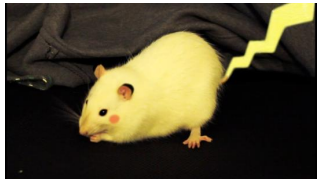


Figure 1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

Domain Adaptation





True

False

Figure 1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

Domain Adaptation

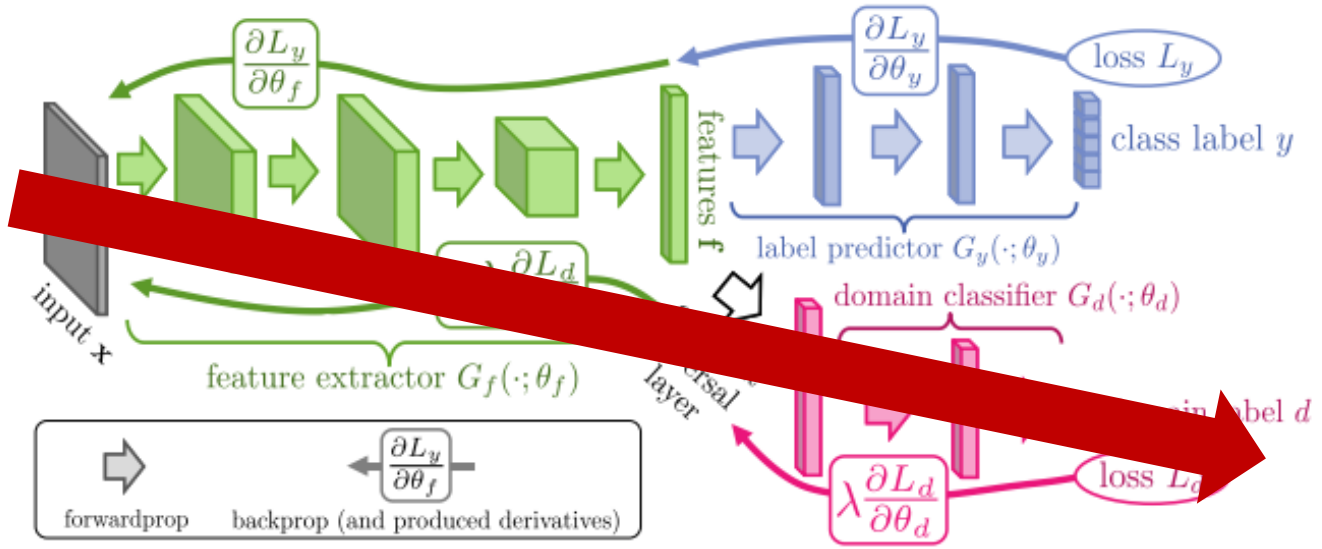
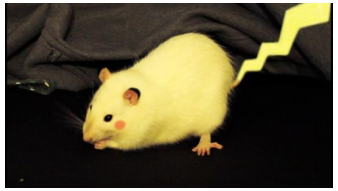


Figure 1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.



Source

Target

Domain Adaptation

Classical vs Adaptation Error

Classical Test Error:

$$\epsilon_{\text{test}} \leq \hat{\epsilon}_{\text{train}} + \sqrt{\frac{\text{complexity}}{n}}$$

Measured on the same distribution!

기존 전략: 최대한 적은 parameter로 training error가 최소인 model을 찾자

Domain Adaptation

Classical vs Adaptation Error

Classical Test Error:

$$\epsilon_{\text{test}} \leq \hat{\epsilon}_{\text{train}} + \sqrt{\frac{\text{complexity}}{n}}$$

Measured on the same distribution!

Adaptation Target Error:

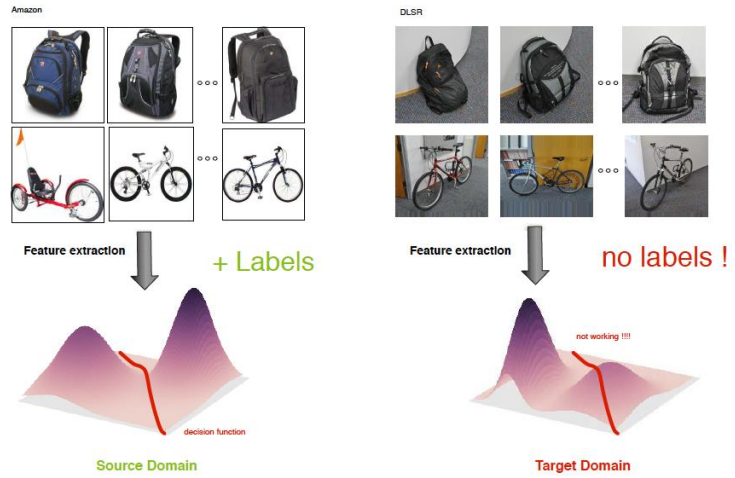
$$\epsilon_{\text{test}} \leq ??$$

이제는 training domain (source)과 testing domain (target)이 서로 다르다

기존의 전략 외에 다른 전략이 추가로 필요하다.

Measured on a **new** distribution!

Domain Shift 고려



Domain Adaptation

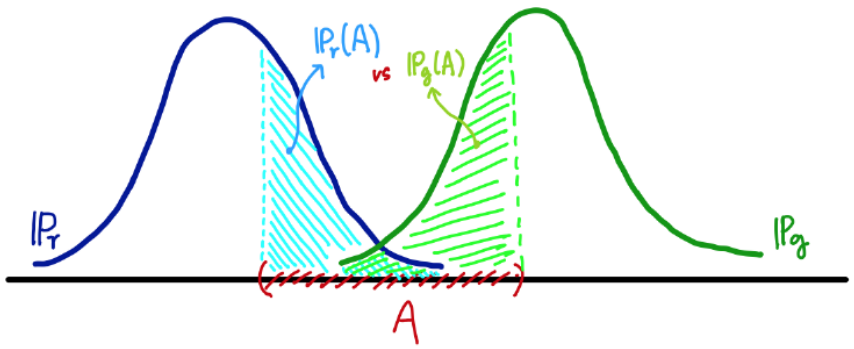
Total Variation (TV)

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$

Total Variation 은 두 확률측도의 측정값이 벌어질 수 있는 값 중 **가장 큰 값** (또는 앞에서 설명한 supremum) 을 말합니다

같은 집합 A 라 하더라도 두 확률분포가 측정하는 값은 다를 수 있습니다. 이 때 TV 는 모든 $A \in \Sigma$ 에 대해 가장 큰 값을 거리로 정의한 겁니다

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$



Domain Adaptation

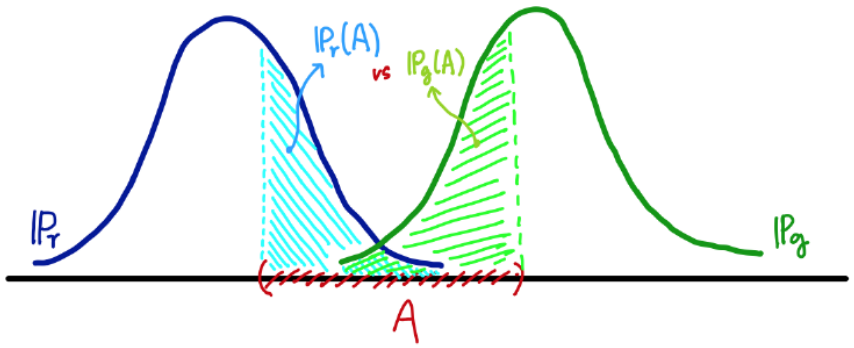
Total Variation (TV)

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$

Total Variation 은 두 확률측도의 측정값이 벌어질 수 있는 값 중 **가장 큰 값**
(또는 앞에서 설명한 supremum) 을 말합니다

같은 집합 A 라 하더라도 두 확률분포가 측정하는 값은 다를 수 있습니다. 이 때 TV 는 모든 $A \in \Sigma$ 에 대해 가장 큰 값을 거리로 정의한 겁니다

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$



Definition 1 (Ben-David et al., 2006, 2010; Kifer et al., 2004) *Given two domain distributions $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ over X, and a hypothesis class $\mathcal{H}$, the* $\mathcal{H}$-divergence *between $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ is*

$$d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^X, \mathcal{D}_{\mathrm{T}}^X) = 2 \sup_{\eta\in\mathcal{H}} \left| \Pr_{\mathbf{x}\sim\mathcal{D}_{\mathrm{S}}^X} [\eta(\mathbf{x}) = 1] - \Pr_{\mathbf{x}\sim\mathcal{D}_{\mathrm{T}}^X} [\eta(\mathbf{x}) = 1] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_{\mathrm{S}}^X$ from examples generated by $\mathcal{D}_{\mathrm{T}}^X$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical* $\mathcal{H}$*-divergence* between two samples $S \sim (\mathcal{D}_{\mathrm{S}}^X)^n$ and $T \sim (\mathcal{D}_{\mathrm{T}}^X)^{n'}$ by computing

$$\hat{d}_{\mathcal{H}}(S,T) = 2\left(1 - \min_{\eta\in\mathcal{H}} \left[\frac{1}{n}\sum_{i=1}^{n} I[\eta(\mathbf{x}_i)=0] + \frac{1}{n'}\sum_{i=n+1}^{N} I[\eta(\mathbf{x}_i)=1]\right]\right), \quad (1)$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

Domain Adaptation

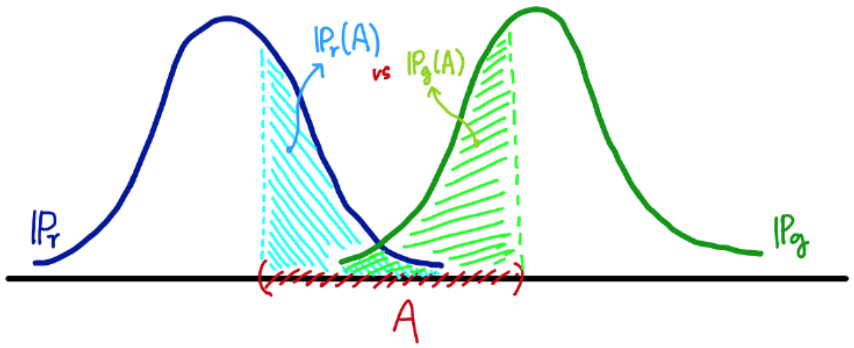
Total Variation (TV)

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$

Total Variation 은 두 확률측도의 측정값이 벌어질 수 있는 값 중 **가장 큰 값**
(또는 앞에서 설명한 supremum) 을 말합니다

같은 집합 A 라 하더라도 두 확률분포가 측정하는 값은 다를 수 있습니다. 이 때 TV 는 모든 $A \in \Sigma$ 에 대해 가장 큰 값을 거리로 정의한 겁니다

$$\delta(\mathbb{P}_r,\mathbb{P}_g) = \sup_{A\in\Sigma} |\mathbb{P}_r(A) - \mathbb{P}_g(A)|$$



Definition 1 (Ben-David et al., 2006, 2010; Kifer et al., 2004) *Given two domain distributions $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ over X, and a hypothesis class $\mathcal{H}$, the* $\mathcal{H}$-divergence *between $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ is*

$$d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^X, \mathcal{D}_{\mathrm{T}}^X) = 2 \sup_{\eta\in\mathcal{H}} \left| \Pr_{\mathbf{x}\sim\mathcal{D}_{\mathrm{S}}^X} [\eta(\mathbf{x}) = 1] - \Pr_{\mathbf{x}\sim\mathcal{D}_{\mathrm{T}}^X} [\eta(\mathbf{x}) = 1] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_{\mathrm{S}}^X$ from examples generated by $\mathcal{D}_{\mathrm{T}}^X$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical* $\mathcal{H}$*-divergence* between two samples $S \sim (\mathcal{D}_{\mathrm{S}}^X)^n$ and $T \sim (\mathcal{D}_{\mathrm{T}}^X)^{n'}$ by computing

$$\hat{d}_{\mathcal{H}}(S,T) = 2\left(1 - \min_{\eta\in\mathcal{H}} \left[\frac{1}{n}\sum_{i=1}^{n} I[\eta(\mathbf{x}_i)=0] + \frac{1}{n'}\sum_{i=n+1}^{N} I[\eta(\mathbf{x}_i)=1]\right]\right), \quad (1)$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

Domain Adaptation

Theorem 2 (Ben-David et al., 2006) *Let $\mathcal{H}$ be a hypothesis class of VC dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^n$ and $T \sim (\mathcal{D}_T^X)^n$, for every $\eta \in \mathcal{H}$:*

$$R_{\mathcal{D}_T}(\eta) \leq R_S(\eta) + \sqrt{\frac{4}{n}\left(d \log \frac{2en}{d} + \log \frac{4}{\delta}\right)} + \hat{d}_{\mathcal{H}}(S,T) + 4\sqrt{\frac{1}{n}\left(d \log \frac{2n}{d} + \log \frac{4}{\delta}\right)} + \beta,$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} [R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)]$, *and*

$$R_S(\eta) = \frac{1}{n}\sum_{i=1}^{m} I\left[\eta(\mathbf{x}_i) \neq y_i\right]$$

A Computable Adaptation Bou

Let $\mathcal{H}$ be a hypothesis class of VC din and $\mathcal{U}_S, \mathcal{U}_T$ be unlabeled samples of size m' each, drawn from $\mathcal{D}_S, \mathcal{D}_T$ respectively. With probability at least $1-\delta$ (over the choice of unlabeled sample), for every $h \in \mathcal{H}$,

$$\epsilon_{\mathcal{D}_T}(h) \leq \epsilon_{\mathcal{D}_S}(h) + \hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) + \lambda + O\left(\sqrt{\frac{d \log \frac{m'}{d} + \log \frac{1}{\delta}}{m'}}\right)$$

Divergence estimation complexity

Dependent on number of unlabeled samples

Penn
University of Pennsylvania

Domain Adaptation

Theorem 2 (Ben-David et al., 2006) *Let $\mathcal{H}$ be a hypothesis class of VC dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_S)^n$ and $T \sim (\mathcal{D}_T^X)^n$, for every $\eta \in \mathcal{H}$:*

Minimize!

$$R_{\mathcal{D}_T}(\eta) \leq R_S(\eta) + \sqrt{\frac{4}{n}\left(d\log\frac{2en}{d} + \log\frac{4}{\delta}\right)} + \hat{d}_{\mathcal{H}}(S,T) + 4\sqrt{\frac{1}{n}\left(d\log\frac{2n}{d} + \log\frac{4}{\delta}\right)} + \beta\,,$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_S}(\eta^*) + R_{\mathcal{D}_T}(\eta^*)\right]$, *and*

$$R_S(\eta) = \frac{1}{n}\sum_{i=1}^{m} I\left[\eta(\mathbf{x}_i) \neq y_i\right]$$

Domain Adaptation

$$
\begin{aligned}
& d_{\mathcal{H}_p \Delta \mathcal{H}_p}(\mathcal{S}, \mathcal{T}) = \\
& = 2 \sup_{h \in \mathcal{H}_p \Delta \mathcal{H}_p} |P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]| \leq \\
& \leq 2 \sup_{h \in \mathcal{H}_d} |P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]| = \\
& = 2 \sup_{h \in \mathcal{H}_d} |1 - \alpha(h)| = 2 \sup_{h \in \mathcal{H}_d} [\alpha(h) - 1]
\end{aligned}
\tag{13}
$$

where $\alpha(h) = P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 0] + P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]$ is maximized by the optimal G_d.

Definition 3 For a hypothesis space $\mathcal{H}$, the *symmetric difference hypothesis space* $\mathcal{H}\Delta\mathcal{H}$ is the set of hypotheses

$\boldsymbol{h \in H \Leftrightarrow 1 - h \in H}$

$$
g \in \mathcal{H}\Delta\mathcal{H} \iff g(\mathbf{x}) = h(\mathbf{x}) \oplus h'(\mathbf{x}) \quad \text{for some } h, h' \in \mathcal{H},
$$

where $\oplus$ is the XOR function. In words, every hypothesis $g \in \mathcal{H}\Delta\mathcal{H}$ is the set of disagreements between two hypotheses in $\mathcal{H}$.

Domain Adaptation

$$
\begin{aligned}
& d_{\mathcal{H}_p \Delta \mathcal{H}_p}(\mathcal{S}, \mathcal{T}) = \\
& = 2 \sup_{h \in \mathcal{H}_p \Delta \mathcal{H}_p} |P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]| \leq \\
& \leq 2 \sup_{h \in \mathcal{H}_d} |P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]| = \\
& = 2 \sup_{h \in \mathcal{H}_d} |1 - \alpha(h)| = 2 \sup_{h \in \mathcal{H}_d} [\alpha(h) - 1]
\end{aligned} \tag{13}
$$

where $\alpha(h) = P_{\mathbf{f} \sim \mathcal{S}}[h(\mathbf{f}) = 0] + P_{\mathbf{f} \sim \mathcal{T}}[h(\mathbf{f}) = 1]$ is maximized by the optimal G_d.

Definition 3 For a hypothesis space $\mathcal{H}$, the *symmetric difference hypothesis space* $\mathcal{H}\Delta\mathcal{H}$ the set of hypotheses

$$\boldsymbol{h \in H \Leftrightarrow 1 - h \in H}$$

$$g \in \mathcal{H}\Delta\mathcal{H} \iff g(\mathbf{x}) = h(\mathbf{x}) \oplus h'(\mathbf{x}) \quad \text{for some } h, h' \in \mathcal{H},$$

where $\oplus$ is the XOR function. In words, every hypothesis $g \in \mathcal{H}\Delta\mathcal{H}$ is the set of disagreements between two hypotheses in $\mathcal{H}$.

Definition 1 (Ben-David et al., 2006, 2010; Kifer et al., 2004) *Given two domain distributions $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ over X, and a hypothesis class $\mathcal{H}$, the* $\mathcal{H}$-divergence *between $\mathcal{D}_{\mathrm{S}}^X$ and $\mathcal{D}_{\mathrm{T}}^X$ is*

$$d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^X, \mathcal{D}_{\mathrm{T}}^X) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x} \sim \mathcal{D}_{\mathrm{S}}^X} [\eta(\mathbf{x}) = 1] - \Pr_{\mathbf{x} \sim \mathcal{D}_{\mathrm{T}}^X} [\eta(\mathbf{x}) = 1] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_{\mathrm{S}}^X$ from examples generated by $\mathcal{D}_{\mathrm{T}}^X$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical* $\mathcal{H}$*-divergence* between two samples $S \sim (\mathcal{D}_{\mathrm{S}}^X)^n$ and $T \sim (\mathcal{D}_{\mathrm{T}}^X)^{n'}$ by computing

$$\hat{d}_{\mathcal{H}}(S, T) = 2 \left(1 - \min_{\eta \in \mathcal{H}} \left[\frac{1}{n} \sum_{i=1}^{n} I[\eta(\mathbf{x}_i) = 0] + \frac{1}{n'} \sum_{i=n+1}^{N} I[\eta(\mathbf{x}_i) = 1] \right] \right), \tag{1}$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

where the examples of the source sample are labeled 0 and the examples of the target sample are labeled 1. Then, the risk of the classifier trained on the new data set U approximates the "min" part of Equation (1). Given a generalization error ϵ on the problem of discriminating between source and target examples, the $\mathcal{H}$-divergence is then approximated by

$$\hat{d}_{\mathcal{A}} = 2(1 - 2\epsilon). \tag{3}$$

Domain Adaptation

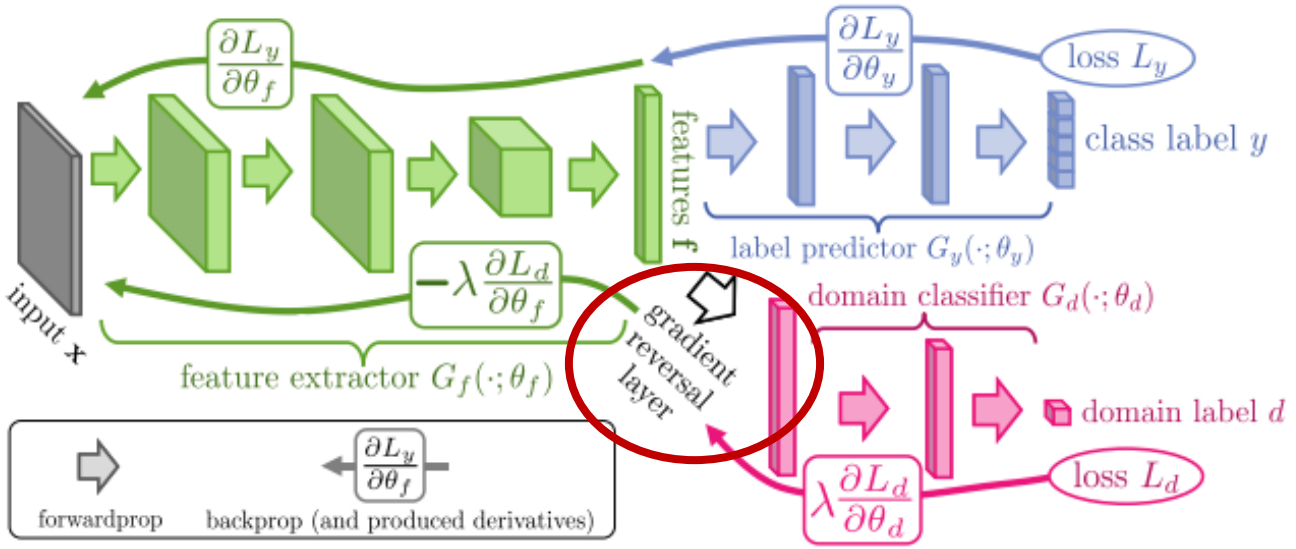
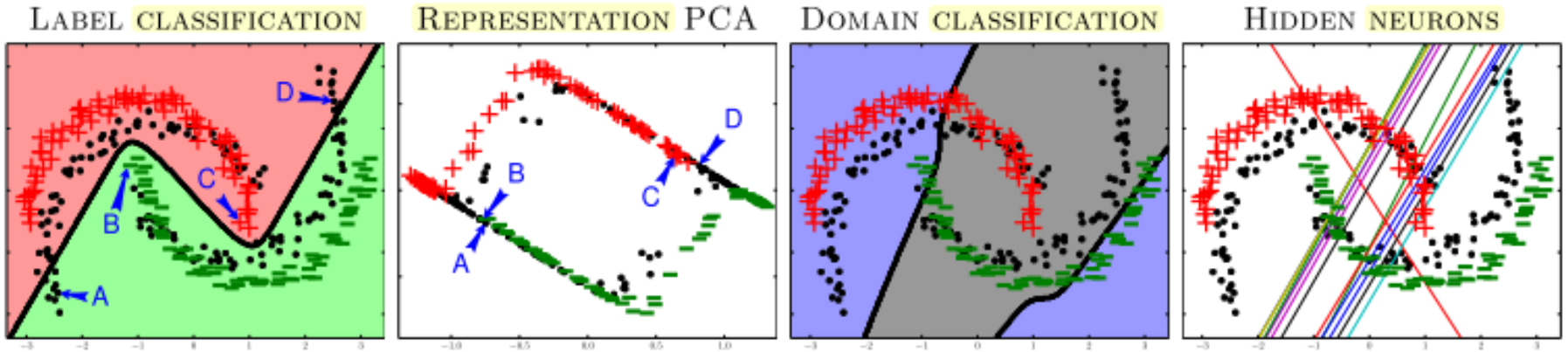
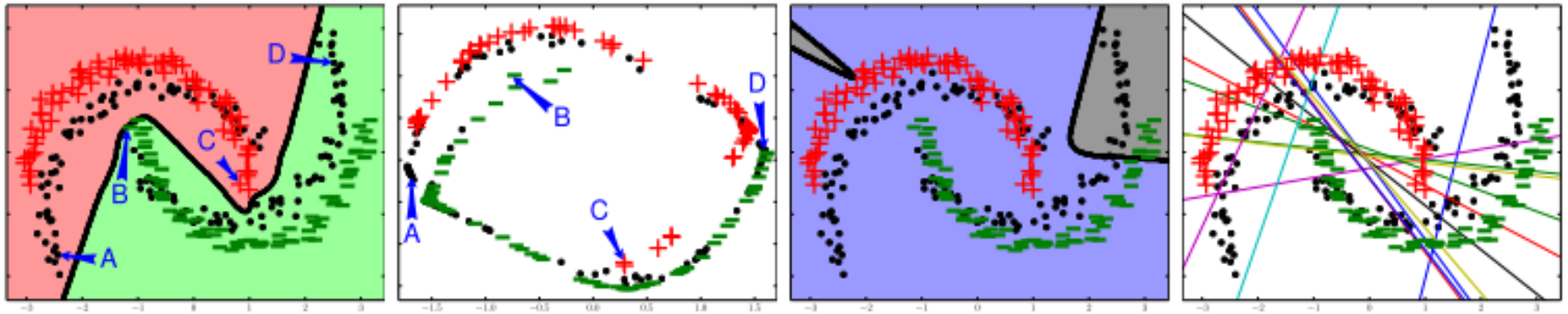


Figure 1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

Domain Adaptation



(a) Standard NN. For the "domain classification", we use a *non adversarial* domain regressor on the hidden neurons learned by the Standard NN. (This is equivalent to run Algorithm 1, without Lines 22 and 31)



(b) DANN (Algorithm 1)

Figure 2: The *inter-twinning moons* toy problem. Examples from the source sample are represented as a "+" (label 1) and a "−" (label 0), while examples from the unlabeled target sample are represented as black dots. See text for the figure discussion.

Reference

https://www.youtube.com/watch?v=n2J7giHrS-Y , PR-12

https://www.slideshare.net/ssuser7e10e4/wasserstein-gan-i , WGAN 수학 이해하기

http://jaejunyoo.blogspot.com/2017/01/domain-adversarial-training-of-neural.html, 유재준님 블로그